VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Absolvování individuální odborné praxe

Individual Professional Practice in the Company

2015 Petra Hudečková

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Petra Hudečková**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Absolvování individuální odborné praxe
Individual Professional Practice in the Company

Zásady pro vypracování:

1. Student vykoná individuální praxi ve firmě: CIT VŠB-TUO
2. Struktura závěrečné zprávy:
   a) Popis odborného zaměření firmy, u které student vykonal odbornou praxi a popis pracovního zařazení studenta.
   b) Seznam úkolů zadaných studentovi v průběhu odborné praxe s vyjádřením jejich časové náročnosti.
   c) Zvolený postup řešení zadaných úkolů.
   d) Teoretické a praktické znalosti a dovednosti získané v průběhu studia uplatněné studentem v průběhu odborné praxe.
   e) Znalosti či dovednosti scházející studentovi v průběhu odborné praxe.
   f) Dosažené výsledky v průběhu odborné praxe a její celkové zhodnocení.

Seznam doporučené odborné literatury:

Podle pokynů konzultanta, který vede odbornou praxi studenta.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **doc. Mgr. Jiří Dvorský, Ph.D.**

Konzultant bakalářské práce: Ing. Martin Lasoň

Datum zadání: 01.09.2014

Datum odevzdání: 07.05.2015

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně. Uvedla jsem všechny literární prameny a publikace, ze kterých jsem čerpala.

V Ostravě 7. května 2015 .....

Ráda bych na tomto místě poděkovala CIT VŠB-TUO za umožnění vykonání bakalářské praxe. Zejména bych chtěla vyjádřit poděkování Ivanu Sokolíkovi, který mi pomáhal a vedl po velkou část doby, dalšímu konzultantovi, ing. Martinu Lasoňovi za užitečné rady a jiným kolegům, jež mi v případě nutnosti dokázali poradit.

Abstrakt

Tato bakalářská práce popisuje mé působení v CIT VŠB-TUO po dobu mého posledního ročníku bakalářského studia. Ve své práci se snažím popsat úkoly, jenž jsem dostala k řešení, nastínit danou problematiku a uvést způsoby řešení a technologie, které jsem při plnění těchto úkolů použila. V závěru je shrnutí nabytých znalostí a výsledků praxe.

Klíčová slova: Bakalářská praxe, CIT VŠB-TUO, Java EE, Hibernate, webová aplikace

Abstract

This bachelor thesis describes my working in CIT VŠB-TUO company for the last year of my bachelor study. In my thesis I try to describe tasks which I had to solve, outline in detail the issues and provide solutions and technologies that I have used in these tasks. I summarise knowledge I've learnt and results of bachelor thesis at the end.

Keywords: bachelor thesis, CIT VŠB-TUO, Java EE, Hibernate, web application

Seznam použitých zkratek a symbolů

Java EE	–	Java Platform, Enterprise Edition
JSP	–	JavaServer Pages
JPA	–	Java Persistence Api
JSTL	–	JavaServer Pages Standard Tag Library
CAS	–	Central Authentication Service
SSO	–	Single sign-on
RSS	–	Rich Site Summary
XML	–	EXtensible Markup Language
JIRA	–	Softwarový nástroj pro usnadnění procesu řízení projektů a požadavků
CSS	–	Cascading Style Sheets
CVS	–	Concurrent Versions System
GIT	–	Distribuovaný systém správy verzí

Obsah

Seznam obrázků

Seznam výpisů zdrojového kódu

1 Úvod

Odbornou praxi jsem se rozhodla absolvovat v CIT VŠB-TUO. Nastoupila jsem tam v době, kdy jsem neuvažovala o možnosti vykonání bakalářské práce místo zadaného tématu, firma ani nebyla v seznamu firem, které praxi umožňovaly. Díky skvělé komunikaci mi to později bylo umožněno, a i přes časovou tíseň kvůli blížejícímu termínu konce změny bakalářské práce vše nakonec vyšlo.

Má očekávání při nástupu byla, že poprvé v životě zažiji práci v týmu, naučím se při tom nové technologie a získám tak zkušenosti, které se pouze při studiu získat nedají.

V této bakalářské práci bych chtěla uvést popis firmy a pracovního zařazení. Dále popíši postupy a technologie použité při řešení jednotlivých úkolů, které jsem v průběhu praxe dostala. V závěru pak následuje zhodnocení, jak se má původní očekávání naplnila.

2 Odborné zařazení firmy a popis pracovního zařazení

2.1 Odborné zaměření firmy

Centrum informačních technologií (CIT) je celoškolské pracoviště Vysoké školy báňské – Technické univerzity Ostrava. Spadá pod Centrum informačních služeb, které je podřízeno rektorovi. CIT se dále dělí na 3 oddělení: Oddělení Informačních systémů (9871), Oddělení infrastruktury IT (9872) a Audiovizuální služby (9873). Jeho posláním je zejména tvorba koncepcí, rozvoj a koordinace nasazení všech typů informačních technologií v rámci VŠB-TUO pro podporu úkolů spojených s předáváním informací svým „zákazníkům“ – studentům, akademickým pracovníkům, zaměstnancům VŠB-TUO, včetně široké veřejnosti. CIT navrhuje koncepci rozvoje informačních technologií na VŠB-TUO, vytváří počítačovou síť univerzity, koordinuje její provoz, zajišťuje technické poradenství pro fakultní a další podsítě, provozuje informační systém univerzity, zabezpečuje jej po HW a SW stránce a poskytuje metodickou pomoc uživatelům systému.

Oddělení informačních systémů se svými 25 pracovníky zajišťuje provoz více než 20 systémů nezbytných pro chod a řízení univerzity, pro podporu výuky, výzkumu a vývoje a spolupráce s praxí. Mezi hlavní informační systémy provozované CIT patří studijní systém EDISON, ekonomický systém SAP, webový portál skládající se z prezentačního webu a portálu pro zaměstnance - InNET, systém pro evidenci publikací OBD, knihovní systém Verbis, digitální repozitář DSpace, Elektronický výukový systém LMS, Elektronický platební systém, systém Evidence projektů, Identity management pro správu uživatelů, kartový systém spolu s pracovištěm kartového centra, Katalog VTS, systém Projekty IRP, systém pro tvorbu rozvrhů Sylabus+, tiskový systém SafeQ, stravovací systém Kredit a několik menších informačních systémů.

2.2 Pracovní zařazení studenta

Již při pohovoru jsem věděla, že bych pracovala jako Java programátor a měla bych na starosti webové aplikace. Po přijetí jsem byla seznámena se všemi službami firmy a bylo mi řečeno, co budu mít na starost.

V začátcích jsem se taktéž seznámila s CVS, systémem správy verzí, a systémem pro správu úkolů, JIRA. Pak jsem začala dělat na prvním ze třech velkých úkolů, které jsem měla v průběhu praxe na starosti.

3 Zadané úkoly a časová náročnost

V průběhu praxe jsem dostala zadané tři hlavní úkoly, které se rozdělily do logických celků. Celkově jsem vytvořila jednu aplikaci. V průběhu úkolů jsem se nedívala na kódy, které tvořily předchozí verze aplikací. Přepsání znamenalo co největší podobnost GUI, zachování funkčnosti, avšak zjednodušení a čistotu kódu.

3.1 Časová náročnost úkolů

Celkový přehled úkolů s podúkoly:

1. Převod dat ze staré verze Profilů
   (a) Návrh a vytvoření databáze: 2 dny
   (b) Přenesení dat do databáze: 5 dnů
2. Přepsání aplikace Profily
   (a) Propojení aplikace s databází: 1 den
   (b) Zobrazení osobní karty: 6 dnů
   (c) Implementace editace dat: 5 dnů
   (d) Testování: 3 dny
   (e) Opravení zobrazování údajů: 2 dny
   (f) Export dat do XML: 3 dny
   (g) Změna způsobu zobrazování jména: 1 den
3. Přepsání aplikace Blog
   (a) Úprava databáze a doménového modelu: 3 dny
   (b) Zobrazování příspěvků: 6 dnů
   (c) Posílání reakcí k blogům: 2 dny
   (d) Vytvoření RSS feedu: 1 den
   (e) Implementace editace blogů: 6 dnů
   (f) Umožnění editace oprávněnou osobou: 4 dnů

3.2 Převod dat ze staré verze Profilů

Tento úkol měl na za cíl změnu uložiště stávajících dat z XML do relační databáze. Musela jsem tedy navrhnout a vytvořit databázi. V další části řešení jsem pak musela převést stávající údaje do nově vytvořené databáze.

3.2.1 Návrh a vytvoření databáze

Vytvoření databáze byl první krok v řešení celého úkolu. Dostala jsem pdf soubor s ukázkovým profilem uživatele, ve kterém byly ukázány všechny údaje, které si uživatel mohl vyplnit. Ten mi sloužil jako výchozí bod k vytvoření návrhu modelu.

Hlavní problémem bylo rozhodnout se, jak naložím s daty. Chtěla jsem využít faktu, že všechny údaje byly rozděleny podle nadpisu. Jednotlivé nadpisy jsem chtěla použít jako názvy tabulek. U některých z nichž však bylo jen málo údajů, tak jsem se je rozhodla spojit k jiným celkům, které byly tématicky podobné. Dalším problémem pak byly dvě jazykové verze každého údaje. Řešením bylo u každé tabulky s údaji atributy s hodnotou české a anglické verze. Složitější řešení alespoň zpočátku nebylo třeba, protože se neočekávalo přibytí další jazykové verze. I zbytek modelu jsem navrhla tak, aby pozdější práce s databází nebyla moc složitá.

Celý návrh modelu měl centrální tabulku s osobou, ke které patřila data. Ostatní údaje byly rozděleny do tématických celků, v nichž každý záznam měl v sobě typ údaje a jeho jazykové hodnoty.

Dále následovalo vytvoření samotného modelu v databázi. Její typ byl pevně daný jako MySQL. Jako první jsem hledala program, který by mi poskytl GUI na práci s databází. Ze školy jsem věděla, že existuje Oracle Developer, který poskytuje uživatelsky přívětivé rozhraní pro práci s databází a umožňuje i její návrh. Ten se však na práci s MySQL nedá použít. Po půlhodině hledání jsem našla MySQL Workbench. V tomto programu se dá graficky vytvořit model databáze a vygenerovat si jeho skript. Této vlastnosti jsem využila, aby nevznikly překlepy a mohla jsem pokračovat v další části úkolu.

3.2.2 Přenesení dat do databáze

Dalším podúkolem byl samotný přenos dat do nově vzniklé databáze. K jeho vyřešení jsem vytvořila program, který dokázal stávající XML data načíst a vygenerovat skript na jejich vložení. Opět jsem na začátku dostala ukázkový soubor, podle kterého jsem měla parser udělat.

Nejprve jsem musela zjistit, jakým způsobem data načtu. Po chvíli hledání jsem našla knihovnu dom4j. Konzultant mi poradil, že můžu iterovat skrze elementy. Já si však vybrala jiný způsob, využití XPath. Ten umožňuje vyhledání elementu na základě výrazu. Přišlo mi to jako výhodnější řešení, protože jsem nepotřebovala hodnoty všech elementů. Pouze jsem se musela naučit jeho syntaxi.

Dalším problémem bylo, že ve starší verzi Profilů se osoba identifikovala na základě jiného údaje, než jaký je v databázi. Na základě tohoto požadavku mi konzultant stvořil CSV soubor, ve kterém byly tyto hodnoty pro každou osobu na jednom řádku

a mě zbývalo je pouze načíst. To mi netrvalo moc dlouho. Jako způsob řešení jsem zvolila ukládání této dvojice hodnot v hash mapě, protože byl vzhledem k počtu záznamu nejefektivnější. Má strategie byla, že při vytváření skriptu pro osobu se vždy zavolá metoda. To by při sekvenčním procházení všech hodnot trvalo dlouho.

Brala jsem v úvahu to, že k přenosu dat dojde pouze jednou a podle toho vytvořila program. Kvůli tomu jsem měla problémy s odlaďováním, kdy jsem už měla skript, ale databáze ho kvůli různým chybám nedokázala vzít.

3.3 Přepsání aplikace Profily

Aplikace Profily převážně sloužila k zobrazení údajů o zaměstnancích VŠB-TUO. Kvůli různým důvodům bylo rozhodnuto, že dojde k jejímu přepsání. Výstupem tedy mělo být zobrazení všech dat o daném člověku a možnost editace těchto údajů. V průběhu řešení pak přibyl i požadavek na zobrazení dat v XML.

Při vytváření aplikace jsem používala CVS, která sloužila k sdílení různých aplikací mezi spolupracovníky. Ze začátku jsem měla problémy, když vznikly konflikty mezi mým a cizím kódem. To se však postupem času spravilo.

3.3.1 Propojení aplikace s databází

Po úspěšném naplnění databáze nastal čas psát samotnou aplikaci. Založila jsem nový Dynamic web project v Eclipse a přemýšlela, jak se bude daná aplikace chovat. To mi moc nešlo, protože jsem neuměla potřebné technologie a i u některých dalších věcích mi nebylo jasné, jak se budou provádět. Věděla jsem pouze, že bude zapotřebí nějakým způsobem se připojit k databázi, provádět databázové operace v nějaké třídě, udělat vrstvu nad tím kvůli přístupu a nakonec zobrazit a umožnit uživateli měnit data.

V rámci příprav bylo zapotřebí server, na kterém nová aplikace poběží. Na rady konzultanta jsem zvolila Tomcat, který mi svými možnostmi postačoval. Zkusila jsem nejprve nejnovější verzi, avšak později jsem kvůli různým problémům nainstalovala o něco starší verzi, kterou používal zbytek týmu.

Jako první jsem vytvořila doménový model. Ten se skládal ze tříd, které měly atributy a názvy stejné jako databázový model. Poté jsem strávila den učením se JPA [2], technologií pro propojení Java objekty a relační databází, které je mezi Java vývojáři velice oblíbené. Na základě znalostí jsem pak anotovala jednotlivé třídy modelu a doufala, že je vše správně. K ověření správnosti bylo zapotřebí ještě teprve nakonfigurovat připojení k databázi a sestavení alespoň jednoduchého dotazu na dabázi.

Vzhledem k tomu, že na pořadí těchto úkolů nezáleželo, jsem se rozhodla sestavit si jednoduchý select pro výběr dat z tabulky. Tato zdánlivě jednoduchá úloha znamenala, že jsem se musela opět věnovat studiu. Tentokrát bylo zapotřebí alespoň na základní úrovni pochopit Hibernate, objektově relační mapování, který mapuje Java objekty a relační databázi. V souvislostí s tím jsem musela i pochopit framework Spring, jenž umožňuje vývojářům zaměřit se spíše na logiku aplikace. Po dnu jsem napsala třídu, která v sobě měla údaje potřebné pro vytváření selectu nad jedné ze tříd doménového modelu. Nad třídou k přístupu k objektům jsem pak udělala i service layer, abych měla jednotný přístup ze všech míst, ze kterých bych potřebovala.

K zobrazování dat mi tedy už chybělo jen samotné připojení k databázi. To spočívalo v zadání správných údajů v jednom souboru. Tato zdánlivě jednoduchá věc mi trvala přes den.

Když se mi podařilo zobrazit data z databáze, bylo zapotřebí zajistit získání i dat, které si uživatelé nemohli měnit, např. jméno. Tyto data se dala nalézt voláním jiné aplikace, která vracela xml. To bylo zapotřebí načíst. K tomu jsem zvolila stejný způsob jako předtím při přenosu dat. Pouze jsem při tom musela ještě řešit případy, kdy byla

duplikátní data. To jsem vyřešila s využitím hash setu, který se duplikátů zbavuje.
U těchto dat jsem musela také vyřešit, v jakém vztahu budou s daty s databází. Nejlepší způsob, který mě napadl, byl ať se data, která se mají v rámci osobní karty zobrazovat pod sebou, ukládají do objektů zvláštní třídy, která má podobnou strukturu jako zbytek modelu.

Pak jsem musela zajistit propojení všech dat. Na to jsem použila servisní vrstvu. Díky té jsem mohla kdekoliv vytvářet osobní karty jednotným voláním metody.

3.3.2 Zobrazení osobní karty

Poté jsem už potřebovala zobrazit data na webové stránce. K tomu jsem potřebovala nastudovat technologii JSP, která umožňuje vytváření dynamicky generovaných stránek HTML. V souvislosti s tím jsem i musela podívat na JSTL [3], které ještě více umožňuje syntézu Javy a HTML.

Kvůli přidání předchozích dat jsem si musela o něco prohloubit znalosti JPA, jelikož mi pak jejich zobrazování končilo chybou. To se naštěstí neukázalo jako velký problém.

Poté nastal čas k zobrazení dat. To se ukázalo být jako vcelku velký problém, protože jsem opět musela dbát na to, že mají být dvě jazykové verze. S tím mi opět pomohlo JSTL. Poté následovalo zobrazování všech dat daného uživatele. Při implementaci mi nastaly dva problémy. Špatně se mi zobrazovaly víceřádková data, k řešení jsem opět mohla využít JSTL. Podruhé jsem musela vyřešit, ať se nezobrazují nadpisy u kategorií, které v sobě nemají žádná data, což jsem vyřešila vytvořením pomocných metod.

Pak bylo zapotřebí stránku nastylovat. Kvůli tomu jsem si musela připomenout CSS. Při zobrazování některých údajů, které byly odkazy a nešlo o manuálně zadaná data, bylo vhodné, aby se při kliknutí neotevřelo nové okno či panel prohlížeče. K tomu dobře posloužil javascriptový plugin prettyphoto. Ten jsem ještě musela ovšem nakonfigurovat tak, aby odpovídal mým představám.

3.3.3 Implementace editace dat

Možnost editace dat byla velmi významným požadavkem na tuto aplikaci. Nejenže bylo zapotřebí zajistit, aby se změněná data ukládala, ale v souvislosti s tím bylo zapotřebí zajistit autentizaci uživatelů.

Nejprve jsem se zaměřila na samotnou možnost editace dat. Udělala jsem formulář, po jehož odeslání se měly změny udělané uživatelem projevit i v databázi. Tak jako předtím jsem použila Hibernate. Ten mě však velice překvapil svým chováním v momentě, kdy jsem ve formuláři měnila nějakou hodnotu a zároveň některou z nich mazala. Po dvou dnech strávených různým konfigurováním persistence jsem poznala, že přestože Hibernate je velice mocný nástroj, který ulehčuje vývojářům práci, je zapotřebí mu rozumět.

Nakonec bylo zapotřebí zabránit, aby lidé měnili cizí profily. K tomu jsem využila již zavedené řešení - autorizace přes SSO, které se odehrává přes CAS filtry. I když samotná konfigurace byla jen zkopírování pár údajů a editace některých věcí, rozhodla jsem se, že se podívám, jakým stylem filtry fungují.

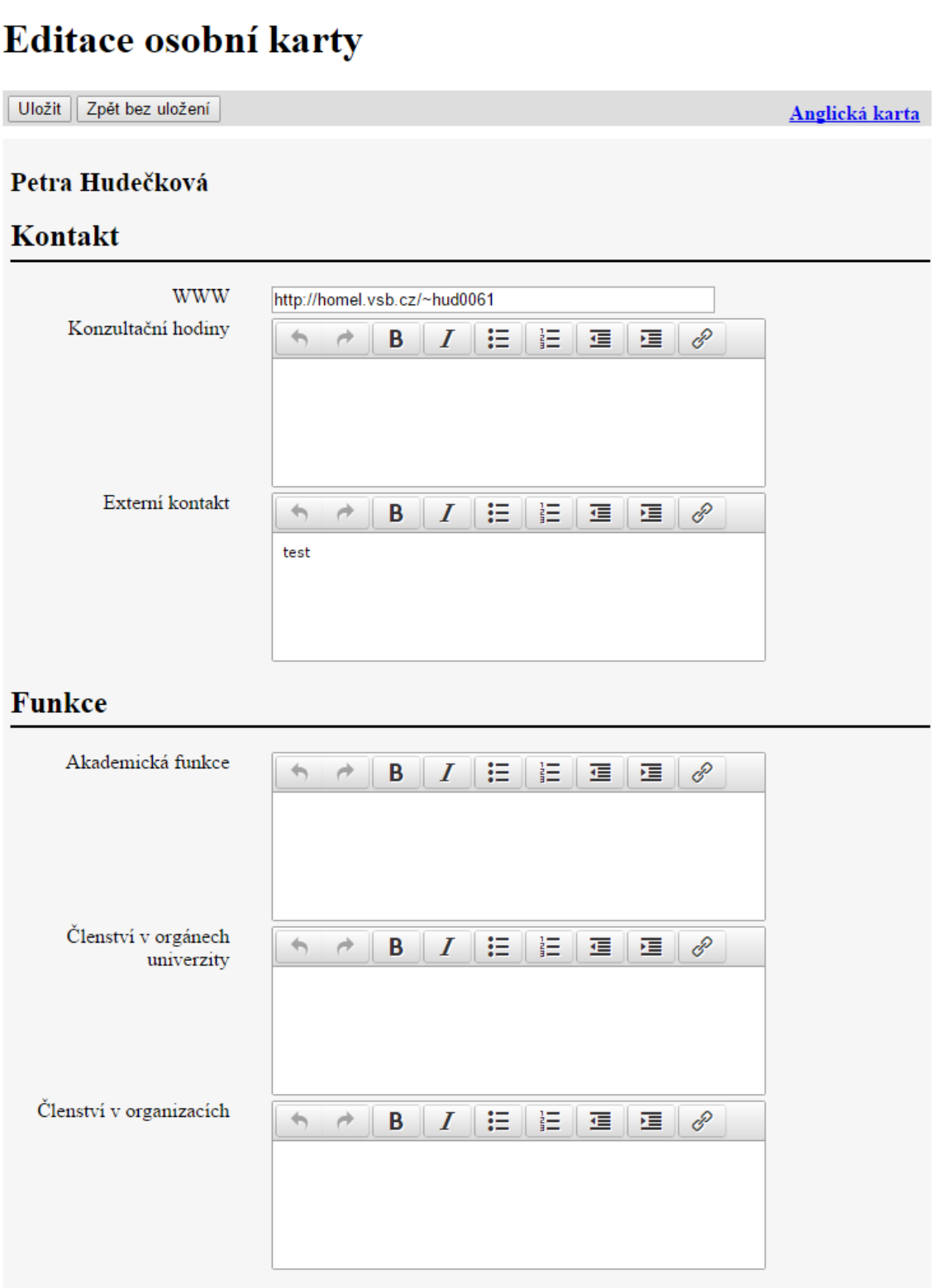

Figure 1: Ukázka editace dat Profilů

3.3.4 Testování

Před nasazením aplikace bylo zapotřebí ji ještě otestovat. Na konci každého podúkolu došlo k otestování očekávané funkčnosti, ale jak jsem už poznala dříve, i když se zdánlivě nic nezměnilo, neznamená to, že se funkčnost nezměnila. Později se zjistilo, že neodhalilo všechny chyby a objevily se problémy.

3.3.5 Opravení zobrazování údajů

Po nasazení aplikace se poměrně brzy objevilo od uživatele hlášení, že se odkazy, které měl uložené ve svých datech v databázi, nejsou klikatelné. Způsob, kterým jsem měla vyřešené zobrazování odkazů počítal pouze s tím, že uživatel má v údajích pouze jeden odkaz. Po napsání SQL dotazu, jsem zjistila, že to nebyla pravda u více uživatelů.

Otázkou bylo co s tím. Dělání víc klikatelných odkazů nemělo smysl, neboť nebylo zřejmé, jakým způsobem bych zjišťovala počet odkazů, jejich začátek a konec. S konzultantem jsme se dohodli, že nejlepší bude u uživatelů, co mají více odkazů, převést odkazy odkazy do jiného typu záznamu.

V souvislosti s tím jsem vyřešila trochu jinou chybu, která však byla součástí daného problému. Při přenosu dat jsem každou položku zakončila '\n'. V editoru dat byl u této položky input, který se s těmito znaky vypořádal stylem, že jej automaticky změnil na <br\>, čehož si mohl uživatel snadno přehlédnout. Tohoto chování jsem se zbavila pomocí metody, která odstraní tyto znaky ještě před zobrazením tohoto údaje.

3.3.6 Export dat do XML

Tento požadavek se objevil až v době, kdy jsem pracovala na dalším úkolu, ale v rámci logiky patří tady. Výsledek měl být prostý - export údajů do XML. Toto pro mě bylo opět něco nového, neboť jsem v Javě nikdy XML nevytvářela. Měla jsem napevno zadaný formát výstupu. Jako vzor mi sloužil zadaný soubor.

Během implementace jsem použila stejnou knihovnu, jakou jsem předtím používala při zobrazení. S její pomocí jsem vytvořila strom znázorňující osobu a dané atributy.

Při jeho vytváření jsem nejprve měla problém přesvědčit JSP, aby XML zobrazovalo v pořádku. Poté, co se mi to podařilo, jsem začala vytvářet XML v zadaném formátu. To se ukázalo být spíše zdlouhavé než složité. Vzhledem k množství chybějících údajů jsem udělala u pár položek chybu. Napravení bylo vzhledem k velice dobře pojmenovaným proměnným jednoduché.

Po vytvoření celého XML jsem zkusila spustit zobrazování. Trvalo přes 4 minuty, než se data objevila. Při nasazení na server to způsobilo time out, jelikož čeká méně času na data. Byly dva způsoby řešení. Buď to bych výstup zobrazovala postupně, ne vše najednou nebo by se výstup periodicky spouštěl a ukládal do souboru, který by byl pro zadavatele přístupný. Jednodušší byl druhý způsob.

3.3.7 Změna způsobu zobrazování jména

Tento podúkol vznikl opět v době, kdy jsem dělala na Blogu. Jeho podstatou bylo zaměnění pořadí křestního jména a přijmení. Většina této změny se udála v jiné aplikaci, pomocí níž si beru tento údaj. Po uvážení všech důsledků bylo lepší udělat nový atribut objektu než pouze předělat ten stávající.

3.4 Přepsání aplikace Blog

Aplikace Blog sloužila k publikování a zobrazování příspěvků. Byla původně samostatná a uživatelé, kteří mohli publikovat, byli pevně dáni. To se však mělo změnit. Na základě podobností s Profily se usoudilo, že by bylo lepší, kdyby se stala její součástí. Toto hledisko pro mě bylo zdánlivě jednodušší, neboť jsem měla vyřešený způsob, kterým bych komunikovala s databází.

3.4.1 Úprava databáze a doménového modelu

Při návrhu třídy a databázové tabulky představující tento podúkol jsem využila stávajícího modelu. Skript na vytvoření tabulky jsem si opět nechala vygenerovat pomocí MySQL Workbench, a pak jsem manuálně udělala pár úprav, které byly zapotřebí. Manuálně jsem přidala foreign key na propojení nově vzniklé tabulky s hlavní tabulkou osoby.

3.4.2 Zobrazování příspěvků

Stejně jako při vytváření předchozí aplikace jsem se rozhodla nejprve zajistit zobrazení dat. Narazila jsem však na velký problém. Ten byl, že se mi nedařilo zobrazit data z více tabulek. Původně jsem si myslela, že jsem špatně napsala JPA anotace ve třídách. Po dni jsem však zjistila, že chyba nebyla v aplikaci, nýbrž v samotné databázi. MySQL Workbench mi sice vygeneroval dobrý skript na vytvoření této tabulky, ale dal tam jiný typ formát uložiště dat, čehož jsem si nevšimla.

Kromě zobrazení jednotlivých příspěvků jsem udělala i postranní sloupeček, ve kterém bylo několik nejnovějších příspěvků. Po kliknutí na starší příspěvky se pak uživatel dostal na stránku, kde byl nadpis a úryvek z obsahu.

3.4.3 Posílání reakcí k blogům

V zadání tohoto úkolu také bylo, ať je možné posílat reakce k jednotlivým blogů uživatelů. To jsem naimpementovala jako formulář, po jehož odeslání se objevilo oznámení pro uživatele. Největší problém byl se zobrazováním tohoto formuláře. Bylo nesmyslné, aby se zobrazoval už při zobrazení příspěvku. Nakonec jsem zvolila k řešení javascript, pomocí něhož se nastavuje viditelnost. Zároveň i při odeslání ověřuji, zda byly vyplněné povinné údaje.

Figure 2: Ukázka zobrazení příspěvku

3.4.4 Vytvoření RSS feedu

V rámci této aplikace jsem měla vytvořit i RSS feed, který by zobrazoval blogy na základě autora. Tento požadavek byl důležitý, neboť i ve starší aplikaci byla tato vlastnost využívaná.

S tvorbou RSS feedu jsem neměla žádné zkušenosti, a tak jsem nejprve zjišťovala, jestli už neexistuje nějaké hotové řešení. Jinak bych musela napsat toto XML sama. Poměrně rychle jsem zjistila, že existuje knihovna, kterou bych mohla použít. Během implementace jsem neměla žádný problém a byla i rychle hotová. Konzultanta pouze zarazilo, že výsledek nevypadal přesně tak, jako bylo u minulé verze. To však bylo způ-

sobené tím, že v předchozí verzi aplikace se používalo RSS 1, zatímco v mém řešení je novější verze, RSS 2.

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:dc="http://purl.org/dc/elements/1.1/" version="2.0">
  <channel>
    <title>Blog - Petra Hudeckova</title>
    <link>http://www.vsb.cz/blog/</link>
    <description>Blog - Petra Hudeckova</description>
    <language>cs</language>
    <dc:language>cs</dc:language>
    <item>
      <title>Test2</title>
      <link>http://www.vsb.cz/blog/cs/HUD0061&id=40</link>
      <description>test</description>
      <pubDate>Mon, 30 Mar 2015 06:14:27 GMT</pubDate>
      <guid>http://www.vsb.cz/blog/cs/HUD0061&id=40</guid>
      <dc:date>2015-03-30T06:14:27Z</dc:date>
    </item>
  </channel>
</rss>
```

Výpis 1: Ukázka RSS feedu

3.4.5 Implementace editace blogů

Implementace možnosti editace příspěvků probíhala velmi podobně jako u Profilů. Základem byl formulář vytváření a editaci, ve kterém bylo možné vložit/změnit příspěvek. Tentokrát jsem však musela použít validaci. To kvůli tomu, aby se mi v databázi neobjevovaly příspěvky, které by měly např. pouze nadpis a žádný další text.

U většiny typů údajů bylo vhodné, aby si člověk mohl vytvářet odkazy a používat stylizaci. Na to však tag <textarea> nebyl vhodný. Bylo mi doporučeno použít TinyMCE, javascriptový editor. Jeho použití mi zpočátku dělalo problémy, avšak po řádném nastavení fungoval, jak měl.

Dále jsem měla umožnit uživatelům přidat obrázek k příspěvkům. Ukládání a následné zobrazování obrázků bylo složitější, než jsem očekávala. Musela jsem změnit typ formuláře a způsob, jakým jsem brala jeho jednotlivé pole. V rámci uživatelské přívětivosti jsem i pomocí javascriptu naimplementovala zobrazení právě nahraného obrázku.

Jako další bylo zapotřebí udělat nějaké uživatelské rozhraní, přes které by se uživatel dostal na vytváření, editaci a mazání příspěvků. Udělala jsem jeho návrh a dala se na implementaci. Má představa byla, že se jako první zobrazí nabídka na vytvoření, a pak seznam všech příspěvků, u kterých bude odkaz na editaci a smazání. Na straně měl být sloupeček s odkazy na nejnovější příspěvky, stejný jako při zobrazování příspěvků. Nejvíce jsem měla potíže s nastylováním.

U mazání příspěvků byl problém, že tam nebyla absolutně žádná ochrana. To bylo nepřijatelné, neboť stačil jediný klik a příspěvek by byl smazán. Jako řešení se naskytovalo použití jen javascriptu, ale to nevypadalo moc dobře. Kolega v tu samou dobu řešil

podobný problém a našel Bootbox, javascriptovou knihovnu. Ta mimo jiné dokáže zobrazit nabídku na potvrzení a vypadá moderně. Napojení na mou aplikaci jsem musela vyřešit sama, kolegův způsob by v mém případě nefungoval.

```
$("#image").change(function(){
    var  file  = document.getElementById("image").files[0];
    var readImg = new FileReader();
    readImg.readAsDataURL(file);
    readImg.onload = function(e) {
      $('.prevImg').attr('src',e.target.result).fadeIn();
    };
 });
```

Výpis 2: Funkce na zobrazení právě nahraného obrázku

Figure 3: Ukázka rozhraní editace blogu

3.4.6 Umožnění editace oprávněnou osobou

U staré verze se stávalo, že vydávaný příspěvek nenapsala přihlášená osoba, nýbrž někdo jiný, kdo ale měl oprávnění. To bylo zapotřebí i tady.

Byly dva způsoby řešení. Jednodušší bylo, kdyby existovala role superadmina, kterou by měli uživatelé, jenž by mohli publikovat za jiné osoby. To však mělo velký nedostatek. Daný uživatel by mohl publikovat za všechny osoby. Druhý způsob řešení tento problém řešil. Šlo o vytvoření databázové tabulky, která by v sobě měla údaj, za koho může nějaký uživatel publikovat. Tento způsob řešení byl složitější na implementaci. Musela jsem předělat část způsobu editování údajů. Při této poměrně velké změně jsem brala v potaz to, aby řešení bylo opět co nejjednodušší a aby dávalo co největší smysl.

Souběžně s tímto úkolem se přecházelo z CVS na Git [5]. V rámci toho bylo vhodné, aby se do CVS nahrály aktuální verze všech aplikací, které měly být v novém Git repozitáři. Musela jsem vygenerovat SSH klíč, pomocí něhož bych se mohla do tohoto repozitáře přihlásit. Oproti kolegům jsem měla výhodu, že na počítači jsem měla Linux, který toto dokáže nativně udělat. Zároveň jsem si tak mohla vyzkoušet i SSH připojení k repozitáři přímo přes konzoli a nemusela používat jiné programy, což je například u Windows potřeba. Celkově přechod na jiný způsob správy verzí aplikací proběhl poměrně hladce.

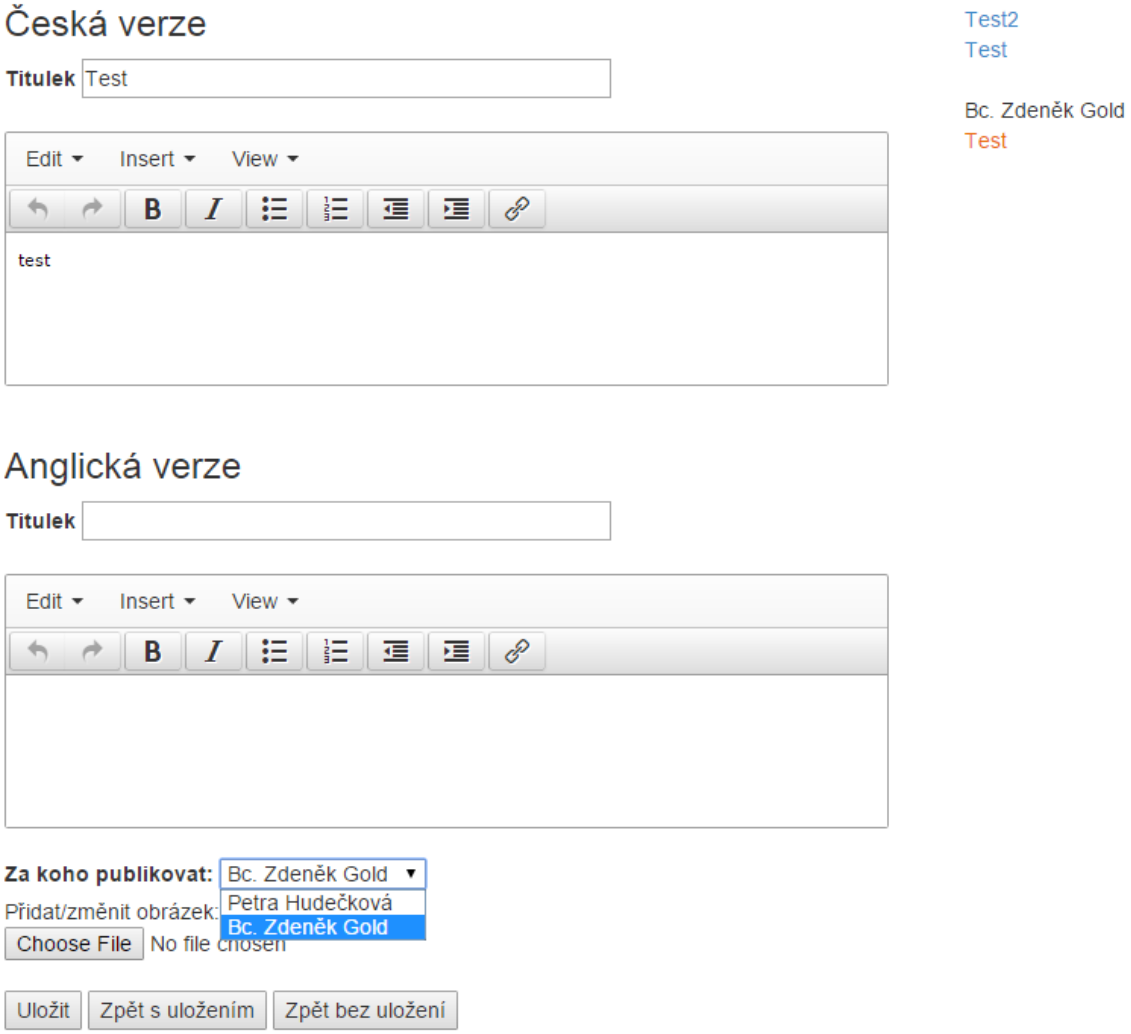

Figure 4: Ukázka editace blogu za jiného uživatele

4 Použité a chybějící znalosti

4.1 Využité předchozí znalosti

V průběhu praxe jsem využila docela mnoho znalostí, co jsem získala ve škole. Již při prvním úkolu jsem použila, co jsem se naučila v Úvodu do databázových systémů při návrhu databáze. Při psaní programu na přesun dat jsem využila znalosti získané z předmětů Algoritmy I a Algoritmy II k tomu, abych měla co nejméně složitý algoritmus a program běžel co nejrychleji. Všechno jsem psala v programovacím jazyku Java, o němž byl předmět Programovací jazyky I.

Při vytváření aplikace jsem pak využila znalosti z Vývoje informačních systémů k správnému rozložení architektury. Díky Databázovým a informačním systémům jsem chápala postup provádění transakcí, což se mi hodilo při upravování dat. Java technologie mi pak pomohly k úplnému pochopení servletů a dalších věcí, kterých se týká Java EE. Vzhledem k tomu, že jsem tvořila aplikace přístupné z webu, potřebovala jsem znalosti HTML, CSS a Javascriptu. Ty mi poskytl předmět Vývoj internetových aplikací. V něm jsem se seznámila i s XPath [4].

Nepřímo jsem i využila znalosti z předmětu Správa operačních systémů, jelikož jsem na pracovním počítači měla distribuci Linuxu a občas jsem musela udělat základní administrátorské úkony, které šly udělat jen v terminálu. Musela jsem tak například vygenerovat si SSH klíče a vyzkoušet si SSH připojení.

4.2 Chybějící znalosti

Zcela nejvíce mi chyběla znalost systémů pro správu verzí, GIT a CVS. Vzhledem k rozšíření Gitu by se hodilo v nějakém předmětu se o něm zmínit a ukázat na klady používání, i když člověk vyvíjí sám.

Dále jsem neměla znalost JSP. Tato technologie se měla učit v předmětu Java technologie, avšak kvůli nedostatku času došlo pouze k jejímu zmínění. Hodilo by se i alespoň základní představení velmi populárního frameworku Spring. Myslím si, že by se o Javě EE mohl udělat celý předmět.

5 Závěr

Za dobu odborné praxe jsem vytvořila aplikaci, která má na starost zobrazování profilů a vytváření blogu. Většina mých očekávání, co jsem měla na začátku, se naplnila. V průběhu praxe jsem se naučila a dozvěděla o hodně nových technologiích, které jsem předtím vůbec neznala nebo jsem s nimi neměla mnoho zkušeností. Tato zkušenost a znalost se mi rozhodně bude hodit i v dalších projektech. Poznala jsem také, jak probíhá práce v týmu a jaká může mít úskalí. Viděla jsem, jak důležitá je vzájemná komunikace, a jak je důležité umět se zeptat, když něco není jasné. Celkově hodnotím svou praxi pozitivně, dala mi toho za poměrně málo času hodně.

Petra Hudečková

6 Literatura

[1] Ray, Eric, *Learning XML, Second Edition*, Sebastopol:O'Reilly Media, 2002.

[2] Yang, Daoqi, *Java Persistence with Jpa 2.1*, Denver:Outskirts Press, 2012.

[3] Geary, David, *Core JSTL: Mastering the JSP Standard Tag Library*, Denver:Prentice Hall, 2002.

[4] Holzner, Steven, *XPath Kick Start: Navigating XML with XPath 1.0 and 2.0*, Indianapolis: Indiana, 2003.

[5] McCullough, Matthew, *Version Control with Git: Powerful tools and techniques for collaborative software development*, Sebastopol:O'Reilly Media, 2012.